МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВВГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6
по дисциплине
«Информатика и программирование»

Студент
гр. БИС-25-3 _____ О.И. Шестопалова
Ассистент
преподавателя _____ М.В. Водяницкий

Владивосток 2025

Задание

Выполнить задания на Python и оформить отчет по стандартам ВВГУ.

Задание 1. Написать функцию, которая конвертирует время из одной величины в другую.

На вход подается:

- число (величина времени)
- исходная единица измерения
- единица измерения, в которую нужно перевести

Функция должна вернуть конвертированное значение

Примеры (формат ввода/вывода можно выбрать свой, если нет строгих требований):

Вход	**Выход**
4h m	240m
30m h	0.5h
12s h	0.03h

Задание 2. Пользователь делает вклад в банке в размере a рублей сроком на n лет

Процент по вкладу зависит от суммы и срока

Зависимость от суммы:

- каждые 10 000 рублей увеличивают ставку на 0.3%
- но суммарное увеличение не может превышать 5%
- минимальный вклад - 30 000 рублей

Зависимость от срока:

- первые 3 года - 3%
- от 4 до 6 лет - 5%
- более 6 лет - 2%

Необходимо написать функцию, которая рассчитывает прибыль пользователя без учета первоначально вложенной суммы

Используется сложный процент: каждый год процент начисляется на текущую сумму вклада

На вход подаются: сумма вклада и количество лет. Результат: сумма прибыли (не весь вклад, а только заработанные проценты)

Примеры:

Вход	Выход
30000 3	3648.67
100000 5	38920.10
200000 8	183925.42

Задание 3. Написать функцию для вывода всех простых чисел в заданном диапазоне. Нужно учитывать некорректные данные (например, начало больше конца или диапазон без простых чисел)

На вход подаются два числа: начало и конец диапазона (включительно). На выходе - список всех простых чисел или сообщение об ошибке

Примеры:

Вход	Выход
1 10	2 3 5 7
15 120	17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113
0 1	Error!

(Формат вывода списка простых чисел может быть любым удобным: в строку через пробел, в несколько строк и т.п.)

Задание 4. Реализовать функцию сложения двух матриц

При сложении двух матриц получается новая матрица того же размера, где каждый элемент - это сумма элементов с тем же индексом из двух исходных матриц

Ограничения:

- складывать можно только матрицы одинакового размера
- размер матрицы должен быть строго больше 2 (например, 3×3, 4×4 и т.д.)
- при нарушении условий нужно вывести сообщение об ошибке

На вход подаются:

1. размер матрицы n (для квадратной матрицы n × n)
2. элементы первой матрицы (по строкам, через пробел)
3. элементы второй матрицы в таком же формате

Результат - новая матрица (в том же формате), либо сообщение об ошибке

Пример (один из возможных вариантов формата):

Вход:

```
2
2 5
5 3
5 2
4 1
```

Выход:

```
7 7
9 4
```

Пример с ошибкой (слишком маленький размер, неправильный ввод и т.п.):

```
1
4
5
```

Выход:

```
Error!
```

Задание 5. Написать функцию, которая определяет, является ли строка палиндромом

Палиндром - это строка, которая читается одинаково слева направо и справа налево (обычно без учета пробелов, регистра и знаков препинания - эти правила нужно явно задать в своей реализации)

На вход подается строка. На выходе:

- Да, если это палиндром
- Нет, если это не палиндром

Примеры:

Вход	**Выход**
А роза упала на лапу Азора	Да
Borrow or rob	Да
Алфавитный порядок	Нет

Содержание

1 Выполнение работы

1.1 Задание 1

На рисунке 1 предоставлен код полученной программы:

```
1   #Задание 1
2
3   def time(x, y, z):
4       unit = {"h": 3600, "m": 60, "s": 1}
5       return x * unit[y] / unit[z]
6   first = time(4, "h", "m")
7   second = time(30, "m", "h")
8   third = time(12, "s", "h")
9   print(f"{first}m")
10  print(f"{second}h")
11  print(f"{third}h")
```

Рисунок 1 — Листинг программы для задания 1

Пояснение работы программы:

1) Объявляем функцию с тремя переменными:

x — кол-во единиц времени

y — исходная единица измерения

z — целевая единица измерения

2) Создаём словарь

3) Переводим x в секунды и делим на кол-во секунд в целевой единице

4) Конвертируем одни единицы измерения в другие

5) Выводим результат

1.2 Задание 2

На рисунке 2 предоставлен код полученной программы:

```
13  #Задание 2
14
15  def calculate(x: int, y: int):
16      if x < 30000:
17          return 0
18      total = 0
19      base = min(0.3 * (x // 10000), 5)
20      for i in range(y):
21          if i <= 2: rate = 3
22          elif i < 6: rate = 5
23          else: rate = 2
24          interest = x * ((base + rate) * 0.01)
25          total += interest
26          x += interest
27      return total
28  first = calculate(30000, 3)
29  second = calculate(100000, 5)
30  third = calculate(200000, 8)
31  print(f"{first}")
32  print(f"{second}")
33  print(f"{third}")
```

Рисунок 2 — Листинг программы для задания 2

Пояснение работы программы:

1) Объявление функции:

x — начальная сумма

y — количество лет

2) Если сумма меньше 30000, то возвращаем к изначальному

3) Создаём переменную для накопления общей суммы процентов

4) Начисление по 0.3% за каждые 10000, но не более 5%

5) Вводим цикл по количеству лет

Если $i \leq 2$ начисляем 3%

Или если $i < 6$ начисляем 5%

В ином случае 2%

6) Вычисляем процент: сумма*общий процент

7) Добавляем к общей сумме процентов

8) Добавляем к общей сумме

9) Возвращаем к общей сумме процентов за весь период

10) Считаем проценты по сумме и по количеству лет

11) Выводим результат

1.3 Задание 3

На рисунке 3 предоставлен код полученной программы:

```
35  #Задание 3
36
37  def prime(n):
38      if n < 2: return False
39      if n == 2: return True
40      if n % 2 == 0: return False
41      return all(n % i for i in range(3, int(n**0.5) + 1, 2))
42  def primes(a, b):
43      p = [str(n) for n in range(a, b + 1) if prime(n)]
44      if not p: return "Error!"
45      r = ""
46      for i in p:
47          r += i + " "
48      return r[:-1]
49  first = primes(1, 10)
50  second = primes(15, 120)
51  third = primes(0, 1)
52  print(f"{first}")
53  print(f"{second}")
54  print(f"{third}")
```

Рисунок 3 — Листинг программы для задания 3

Пояснение работы программы:

1) Объявляем функцию с одной переменной (простые числа)

2) Если x < 2, то число составное

Если x = 2, то число простое

Если x кратно 2, то число составное

3) Возвращаем значение, если все проверки выполняются

4) Объявляем функцию с диапазонов от a до b

5) Создаём список простых чисел в диапазоне

6) Проверяем функцию для каждого x

7) Если проверка успешна, то преобразуем в строку и отправляем в список

8) Если список пустой, то выводим "Error!"

9) Собираем результаты в строку и записываем их через пробел

10) Возвращаем строчку без последнего пробела

11) Присваиваем строчке интервал

12) Выводим результат

1.4 Задание 4

На рисунке 4 предоставлен код полученной программы:

```
#Задание 4

def matrix(n):
    return [list(map(int, input().split())) for _ in range(n)]
def summa(x, y, z):
    for i in range(n):
        for j in range(n):
            print(x[i][j] + y[i][j], end=" ")
        print()
n = int(input())
first = matrix(n)
second = matrix(n)
if any(len(row) != n for row in first + second):
    print("Error!")
else:
    summa(first, second, n)
```

Рисунок 4 — Листинг программы для задания 4

Пояснение работы программы:

1) Объявляем функцию с одной переменной

1) Возвращаем к тому, что просим пользователя ввести целочисленное число (порядок матрицы)

2) Создаём последовательность от 0 до x-1

3) Просим пользователя ввести целые числа (строки матрицы)

4) split читает строчку и разбирает её на список подстрок по пробелам

5) Преобразуем результат в список

6) Объединяем стоки двух матриц в один список

7) Объявляем функцию с тремя переменными

8) Складываем соответствующие элементы списка (матрицы)

9) Выводим результат в квадратном виде (через пробел)

10) Проверяем в цикле строки является ли её длина x

11) Цикл продолжается пока хотя бы один элемент будет равен не x

12) Если не x, то выводим "Error!"

13) В ином случае вызываем функцию

1.5 Задание 5

На рисунке 5 предоставлен код полученной программы:

```python
#Задание 5

def palindrome(text):
    symbols = " !?,.;:-_'\""
    list = {
        'A': 'a', 'B': 'b', 'C': 'c', 'D': 'd', 'E': 'e', 'F': 'f', 'G': 'g',
        'H': 'h', 'I': 'i', 'J': 'j', 'K': 'k', 'L': 'l', 'M': 'm', 'N': 'n',
        'O': 'o', 'P': 'p', 'Q': 'q', 'R': 'r', 'S': 's', 'T': 't', 'U': 'u',
        'V': 'v', 'W': 'w', 'X': 'x', 'Y': 'y', 'Z': 'z',
        'А': 'а', 'Б': 'б', 'В': 'в', 'Г': 'г', 'Д': 'д', 'Е': 'е', 'Ё': 'ё',
        'Ж': 'ж', 'З': 'з', 'И': 'и', 'Й': 'й', 'К': 'к', 'Л': 'л', 'М': 'м',
        'Н': 'н', 'О': 'о', 'П': 'п', 'Р': 'р', 'С': 'с', 'Т': 'т', 'У': 'у',
        'Ф': 'ф', 'Х': 'х', 'Ц': 'ц', 'Ч': 'ч', 'Ш': 'ш', 'Щ': 'щ', 'Ъ': 'ъ',
        'Ы': 'ы', 'Ь': 'ь', 'Э': 'э', 'Ю': 'ю', 'Я': 'я'
    }
    y = ""
    for x in text:
        if x not in symbols:
            y += list.get(x, x)
    return "Да" if y == y[::-1] else "Нет"
first = palindrome("А роза упала на лапу Азора")
second = palindrome("Borrow or rob")
third = palindrome("Алфавитный порядок")
print(f"{first}")
print(f"{second}")
print(f"{third}")
```

Рисунок 5 — Листинг программы для задания 5

Пояснение работы программы:

1) Объявляем функцию с одной переменной

text — строчка для проверки

2) Создаём строчку с символами

3) Создаём словарь

4) Создаём пустую переменную

5) Создаём цикл для x, если x не содержит символов

6) Добавляем к пустой переменной словарь, в котором заглавные буквы возвращаются к строчным

7) Если строчка одинакова с перевёрнутой (палиндром), то выводим "Да", иначе "Нет"

8) Присваиваем фразу строчке

9) Выводим результат